CODI: Combinatorial Optimization for Data Integration – Results for OAEI 2010

Jan Noessner and Mathias Niepert

KR & KM Research Group
University of Mannheim, Germany
{jan, mathias}@informatik.uni-mannheim.de

Abstract. The problem of linking entities in heterogeneous and decentralized data repositories is the driving force behind the data and knowledge integration effort. In this paper, we describe our probabilistic-logical alignment system CODI (Combinatorial Optimization for Data Integration). The system provides a declarative framework for the alignment of individuals, concepts, and properties of two heterogeneous ontologies. CODI leverages both logical schema information and lexical similarity measures with a well-defined semantics for A-Box and T-Box matching. The alignments are computed by solving corresponding combinatorial optimization problems.

1 Presentation of the system

1.1 State, purpose, general statement

CODI (**C**ombinatorial **O**ptimization for **D**ata **I**ntegration) leverages terminological structure for ontology matching. The current implementation produces mappings between concepts, properties, and individuals including mappings between object and data type properties. The system combines lexical similarity measures with schema information to reduce or completely avoid *incoherence* and *inconsistency* during the alignment process. The system is based on the syntax and semantics of Markov logic [2] and transforms the alignment problem to a maximum-a-posteriori optimization problem.

1.2 Specific techniques used

Markov logic combines first-order logic and undirected probabilistic graphical models [11]. A Markov logic network (MLN) is a set of first-order formulae with weights. Intuitively, the more evidence there is that a formula is true the higher the weight of this formula. It has been proposed as a possible approach to several problems occurring in the context of the semantic web [2]. We have shown that Markov logic provides a suitable framework for ontology matching as it captures both *hard* logical axioms and *soft* uncertain statements about potential correspondences between entities. The probabilistic-logical framework we propose for ontology matching essentially adapts the syntax and semantics of Markov logic. However, we always *type* predicates and we require a strict distinction between *hard* and *soft* formulae as well as *hidden* and *observable* predicates. Given a set of constants (the classes and object properties of the

ontologies), a set of formulae (the axioms holding between the objects and classes), and confidence values for correspondences, a Markov logic network defines a probability distribution over possible alignments. We refer the reader to [9, 8] for an in-depth discussion of the approach and some computational challenges. For generating the Marcov logic networks we used the approach described in [12].

T-Box Matching Formalization Given two ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$ and an initial a-priori similarity measure σ we apply the following formalization. First, we introduce observable predicates O to model the structure of $\mathcal{O}_1$ and $\mathcal{O}_2$ with respect to both concepts and properties. For the sake of simplicity we use uppercase letters D, E, R to refer to individual concepts and properties in the ontologies and lowercase letters d, e, r to refer to the corresponding constants in C. In particular, we add ground atoms of observable predicates to $\mathcal{F}^h$ for $i \in \{1, 2\}$ according to the following rules[1]:

$$\mathcal{O}_i \models D \sqsubseteq E \mapsto sub_i(d, e)$$
$$\mathcal{O}_i \models D \sqsubseteq \neg E \mapsto dis_i(d, e)$$
$$\mathcal{O}_i \models \exists R.\top \sqsubseteq D \mapsto sub_i^d(r, d)$$
$$\mathcal{O}_i \models \exists R.\top \sqsupseteq D \mapsto sup_i^d(r, d)$$
$$\mathcal{O}_i \models \exists R.\top \sqsubseteq \neg D \mapsto dis_i^d(r, d)$$

The ground atoms of observable predicates are added to the set of hard constraints $\mathcal{F}^h$, forcing them to hold in computed alignments. The hidden predicates m_c and m_p, on the other hand, model the sought-after concept and property correspondences, respectively. Given the state of the observable predicates, we are interested in determining the state of the hidden predicates that maximize the a-posteriori probability of the corresponding possible world. The ground atoms of these hidden predicates are assigned the weights specified by the a-priori similarity σ. The higher this value for a correspondence the more likely the correspondence is correct *a-priori*. Hence, the following ground formulae are added to $\mathcal{F}^s$:

$(m_c(c, d),\ \ \sigma(C, D))$	if C and D are concepts
$(m_p(p, r),\ \ \sigma(P, R))$	if P and R are properties

Notice that the distinction between m_c and m_p is required since we use typed predicates and distinguish between the *concept* and *property* type.

Cardinality Constraints A method often applied in real-world scenarios is the selection of a functional one-to-one alignment [1]. Within the ML framework, we can include a set of hard cardinality constraints, restricting the alignment to be functional and one-to-one. In the following we write x, y, z to refer to variables ranging over the appropriately typed constants and omit the universal quantifiers.

$$m_c(x, y) \wedge m_c(x, z) \Rightarrow y = z$$
$$m_c(x, y) \wedge m_c(z, y) \Rightarrow x = z$$

Analogously, the same formulae can be included with hidden predicates m_p, restricting the property alignment to be one-to-one and functional.

[1] Due to space considerations the list is incomplete. For instance, predicates modeling range restrictions are not included.

Coherence Constraints Incoherence occurs when axioms in ontologies lead to logical contradictions. Clearly, it is desirable to avoid incoherence during the alignment process. All existing approaches to alignment repair remove correspondences after the computation of the alignment. Within the ML framework we can incorporate incoherence reducing constraints *during* the alignment process for the first time. This is accomplished by adding formulae of the following type to $\mathcal{F}^h$.

$$dis_1(x, x') \wedge sub_2(x, x') \Rightarrow \neg(m_c(x, y) \wedge m_c(x', y'))$$
$$dis_1^d(x, x') \wedge sub_2^d(y, y') \Rightarrow \neg(m_p(x, y) \wedge m_c(x', y'))$$

Stability Constraints Several approaches to schema and ontology matching propagate alignment evidence derived from structural relationships between concepts and properties. These methods leverage the fact that existing evidence for the equivalence of concepts C and D also makes it more likely that, for example, child concepts of C and child concepts of D are equivalent. One such approach to evidence propagation is *similarity flooding* [7]. As a reciprocal idea, the general notion of stability was introduced, expressing that an alignment should not introduce new structural knowledge [5]. The *soft* formula below, for instance, decreases the probability of alignments that map concepts X to Y and X' to Y' if X' subsumes X but Y' does *not* subsume Y.

$$(sub_1(x, x') \wedge \neg sub_2(y, y') \Rightarrow m_c(x, y) \wedge m_c(x', y'),\ w_1)$$
$$(sub_1^d(x, x') \wedge \neg sub_2^d(y, y') \Rightarrow m_p(x, y) \wedge m_c(x', y'),\ w_2)$$

Here, w_1 and w_2 are *negative* real-valued weights, rendering alignments that satisfy the formulae possible but less likely.

The presented list of cardinality, coherence, and stability constraints could be extended by additional soft and hard formulae. Other constraints could, for example, model known correct correspondences or generalize the one-to-one alignment to m-to-n alignments.

A-Box Matching The current instance matching configuration of CODI leverages terminological structure and combines it with lexical similarity measures. The approach is presented in more detail in [10]. It uses one T-Box $\mathcal{T}$ but two different A-Boxes $\mathcal{A}_1 \in \mathcal{O}_1$ and $\mathcal{A}_2 \in \mathcal{O}_2$. In cases with two different T-Boxes the T-Box matching approach is applied as a preprocessing step, merge the two aligned T-Boxes and then use our instance matching algorithm. CODI offers complete conflict elimination meaning that the resulting alignment is always coherent for OWL DL ontologies. This component is based on the work of Meilicke et al. [6]. CODI enforces the instance alignment to be consistent. To this end, we need to introduce observable predicates O to model conflicts, that is, a positive assertion of one instance in one ontology and a negative assertion of the same instance in the other ontology. This is done for both property and concept assertions.

Analogous to the concept and property alignment before, we introduce the hidden predicate m_i representing instance correspondences. Let C be a concept and P be a property of T-Box $\mathcal{T}$. Further, let $A \in \mathcal{A}_1$ and $B \in \mathcal{A}_2$ be individuals in the respective A-Boxes. Then, using a reasoner, ground atoms are added to the set of *hard* constraints

$\mathcal{F}^h$ according to the following rules:

$$\begin{aligned}
\mathcal{T} \cup \mathcal{A}_1 \models C(A) \wedge \mathcal{T} \cup \mathcal{A}_2 \models \neg C(B) &\mapsto \neg m_i(a,b) \\
\mathcal{T} \cup \mathcal{A}_1 \models \neg C(A) \wedge \mathcal{T} \cup \mathcal{A}_2 \models C(B) &\mapsto \neg m_i(a,b) \\
\mathcal{T} \cup \mathcal{A}_1 \models P(A,A') \wedge \mathcal{T} \cup \mathcal{A}_2 \models \neg P(B,B') &\mapsto \neg m_i(a,b) \vee \neg m_i(a',b') \\
\mathcal{T} \cup \mathcal{A}_1 \models \neg P(A,A') \wedge \mathcal{T} \cup \mathcal{A}_2 \models P(B,B') &\mapsto \neg m_i(a,b) \vee \neg m_i(a',b')
\end{aligned}$$

In addition to these formulae we included cardinality constraints analogous to those used in the concept and property matching of Section 1.2. In the instance matching formulation, the a-priori similarity $\sigma_{\mathbf{c}}$ and $\sigma_{\mathbf{p}}$ measures the *normalized overlap* of concept and property assertions, respectively. For more details on these measures, we refer the reader to [10]. The following formulae are added to the set of soft formulae $\mathcal{F}^s$:

$$\begin{aligned}
(m_i(a,b),\ \ \sigma_{\mathbf{c}}(A,B)) &\quad \text{if A and B are instances} \\
(m_i(a,b) \wedge m_i(c,d),\ \ \sigma_{\mathbf{p}}(A,B,C,D)) &\quad \text{if A, B, C, and D are instances}
\end{aligned}$$

1.3 Adaptations made for the evaluation

The strength of the system is its modularity allowing the incorporation of different similarity measures. The system can be optimized in two major ways: (a) Inclusion of novel formulae enforcing the logical consistency and (b) the inclusion of additional similarity measures. There is room for improvement since we used a very simple lexical similarity measure based on the Levenshtein distance [4] for our experiments. It is possible to apply different aggregation functions like average or maximum and to include specific properties of an ontology like URIs, labels, and comments.

In all OAEI test cases Algorithm 1 was used for computing the a-priori similarity $\sigma(entity_1, entity_2)$. In the case of concept and property alignments, the a-priori similarity is computed by taking the maximal similarity between the URIs, labels and *OBO to OWL* constructs. In case of instance matching the algorithm goes through all data properties and takes the average of the similarity scores.

1.4 Link to the System and Parameters File

CODI can be downloaded from `http://codi-matcher.googlecode.com`.

1.5 Link to the Set of Provided Alignments

The alignments for the tracks *Benchmark* and *Conference* has been made with the SEALS platform. For *Anatomy*, *IIMB*, and *Restaurant* the alignments can be found at `http://code.google.com/p/codi-matcher/downloads/list`

2 Results

In the following section, we present the results of the CODI system for the individual OAEI tracks. Due to space considerations, we do not explain the different benchmarks in more detail.

Algorithm 1 $\sigma(entity_1, entity_2)$

if $entity_1$ and $entity_2$ are either concepts or properties **then**
  $value \leftarrow 0$
  for all Values s_1 of URI, labels, and OBOtoOWL constructs in $entity_1$ **do**
    for all Values s_2 of URI, labels, and OBOtoOWL constructs in $entity_1$ **do**
      $value \leftarrow Max(value, sim(s_1, s_2))$
    end for
  end for
  return $value$
end if
if $entity_1$ and $entity_2$ are individuals **then**
  $Map\langle URI, double\rangle\ similarities \leftarrow null$
  for all dataproperties dp_1 of $entity_1$ **do**
    $uri_1 \leftarrow$ URI of dp_1
    for all dataproperties dp_2 of $entity_2$ **do**
      if uri_1 equals URI of dp_2 **then**
        $value \leftarrow sim(valueofdp_1, valueofdp_2)$
        if uri_1 is entailed in $similarities$ **then**
          update entry $\langle uri_1, old\_value\rangle$ to $\langle uri_1,$ Minimum $(old\_value + value, 1)\rangle$ in $similarities$
        else
          add new entry pair $\langle uri1, value\rangle$ in $similarities$
        end if
      end if
    end for
  end for
  return (sum of all values in $similarities$)/(length of $similarities$)
end if

Benchmark Track While our system's strength is its modularity and adaptability to different ontologies we used the *exact same setting* for all ontology matching tracks. Hence, the performance on the *benchmark* track is rather poor. This is primarily due to the high threshold of 0.85 for the Levenshtein similarity measure that we applied in each of the ontology matching tracks. The results are shown in Table 1.

Table 1. Benchmark results

	1xx	2xx	3xx	Average
Precision	1	0.70	0.92	0.72
Recall	0.99	0.42	0.43	0.44
F_1 score	1	0.49	0.56	0.51

Conference Track On the real-world conference dataset CODI achieves very good results since it employs logical reasoning to avoid incoherences. The execution time is between 2 and 4 minutes per test case[2]. Table 2 summarizes the overall results.

Table 2. Conference results

	Average
Precision	0.87
Recall	0.51
F_1 score	0.64

Anatomy Track The results on the anatomy track are also convincing. The results shown in Table 3 are en par with the 2009 results of state-of-the-art matching applications. The F_1 scores are between 0.79 and 0.73 for all subtasks, even for the two tasks *Focus on Precision* and *Focus on Recall*. Thus, our algorithm achieves satisfiable precision and recall values without sacrifices on the F_1 score. For the last task, where a partial reference alignment was given, we could gain almost 5 % on the F_1 score. This is because incorporating a partial reference alignment in our system is straight-forward. The reference alignment becomes a direct part of the optimization problem, enforcing good correspondences while ruling out contradicting ones. However, since our algorithm uses logical reasoning and has to solve an NP-hard optimization problem, the execution times are quite high[3].

Table 3. Anatomy results

	Focus on F_1 score	Focus on Precision	Focus on Recall	Partial Alignment
Precision	0.954	0.964	0.782	0.969
Recall	0.680	0.663	0.695	0.742
F_1 score	0.794	0.784	0.736	0.840
Execution Time (min)	88	60	157	95

[2] All experiments are executed on a Desktop PC with 2 GB RAM and a Intel Core2 Duo 2.4 GHz processor.

[3] This forces us to submit the solutions without the seals platform because of a timeout after 45 minutes.

IIMB Track The instance matching benchmark IIMB consists of 80 transformations divided in four transformation categories containing 20 transformations each. We applied the full A-Box matching functionality described above with a threshold on the a-priori similarity of 0.1. The average execution time on the IIMB small (large) dataset is 2.6 (35.1) minutes. Table 4 summarizes the different results of the CODI system. The values without brackets are the results for the small IIMB dataset and the values in brackets for the large one.

Table 4. IIMB results

Transformations	0-20	21-40	41-60	61-80[4]	overall
Precision	0.99 (0.98)	0.95 (0.94)	0.96 (0.99)	0.86 (0.86)	0.94 (0.95)
Recall	0.93 (0.87)	0.83 (0.79)	0.97 (0.99)	0.54 (0.53)	0.83 (0.80)
F_1 score	0.96 (0.91)	0.88 (0.85)	0.97 (0.99)	0.65 (0.63)	0.87 (0.85)

PR Track For this track consisting of small files about persons and restaurants, we used a simple one to one alignment only based on lexical similarity scores since no significant structural information is available. Thus, the runtime was with less than 5 seconds per test case very short. The results of the CODI system are depicted in Table 5.

Table 5. PR results

	Person1	Person2	Restaurant
Precision	0.87	0.83	0.71
Recall	0.96	0.22	0.72
F_1-score	0.91	0.36	0.72

3 General comments

3.1 Discussions on the way to improve the proposed system

CODI is a very young system and does not yet provide a user interface. Hence, improvements in usability by designing a suitable user interface will be one of the next steps. In case of the quality of the alignments, more sophisticated lexical similarity measures will be tested and integrated. We are also working on novel algorithms solving the optimization problems more efficiently.

3.2 Comments on the OAEI 2010 procedure

The SEALS evaluation campaign is very beneficial since it is the first time that the matchers must have a standardized interface which could possibly be used by everyone.

3.3 Comments on the OAEI 2010 measures

We encorage the organizers to use semantic precision and recall measures as described in [3].

4 Conclusion

CODI performs concept, property, and instance alignments. It combines logical and structural information with a-priori similarity measures in a well-defined way by using the syntax and semantics of Markov logic. The system therefore not only aligns the entities with the highest lexical similarity but also enforces the coherence and consistency of the resulting alignment.

The overall results of the young system are very promising. Especially when considering the fact that there are many optimization possibilities with respect to the lexical similarity measures that have not yet been investigated. The strength of the CODI system is the combination of lexical and structural information and the declarative nature that allows easy experimentation. We will continue the development of the CODI system and hope that our approach inspires other researchers to leverage terminological structure for ontology matching.

References

1. I. Cruz, F. Palandri, Antonelli, and C. Stroe. Efficient selection of mappings and automatic quality-driven combination of matching methods. In *Proceedings of the ISWC 2009 Workshop on Ontology Matching*, 2009.
2. P. Domingos, D. Lowd, S. Kok, H. Poon, M. Richardson, and P. Singla. Just add weights: Markov logic for the semantic web. In *Proceedings of the Workshop on Uncertain Reasoning for the Semantic Web*, pages 1–25, 2008.
3. D. Fleischhacker and H. Stuckenschmidt. A Practical Implementation of Semantic Precision and Recall. In *2010 International Conference on Complex, Intelligent and Software Intensive Systems*, pages 986–991. IEEE, 2010.
4. V. I. Levenshtein. Binary codes capable of correcting deletions and insertions and reversals. *Doklady Akademii Nauk SSSR*, pages 845–848, 1965.
5. C. Meilicke and H. Stuckenschmidt. Analyzing mapping extraction approaches. In *Proceedings of the Workshop on Ontology Matching*, Busan, Korea, 2007.
6. C. Meilicke, A. Tamilin, and H. Stuckenschmidt. Repairing ontology mappings. In *Proceedings of the Conference on Artificial Intelligence*, pages 1408–1413, Vancouver, Canada, 2007.
7. S. Melnik, H. Garcia-Molina, and E. Rahm. Similarity flooding: A versatile graph matching algorithm and its application to schema matching. In *Proceedings of ICDE*, pages 117–128, 2002.
8. M. Niepert. A Delayed Column Generation Strategy for Exact k-Bounded MAP Inference in Markov Logic Networks. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, 2010.
9. M. Niepert, C. Meilicke, and H. Stuckenschmidt. A Probabilistic-Logical Framework for Ontology Matching. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence*, 2010.
10. J. Noessner, M. Niepert, C. Meilicke, and H. Stuckenschmidt. Leveraging Terminological Structure for Object Reconciliation. *The Semantic Web: Research and Applications*, pages 334–348, 2010.
11. M. Richardson and P. Domingos. Markov logic networks. *Machine Learning*, 62(1-2):107–136, 2006.
12. S. Riedel. Improving the accuracy and efficiency of map inference for markov logic. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 468–475, 2008.